

Software Product Lines: Reuse That Makes Business Sense

Linda Northrop

ASWEC 2006



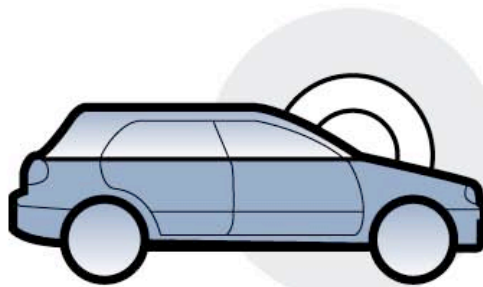
Software Engineering Institute

CarnegieMellon

© 2006 Carnegie Mellon University

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE APR 2006		2. REPORT TYPE		3. DATES COVERED 00-00-2006 to 00-00-2006	
4. TITLE AND SUBTITLE Software Product Lines: Reuse That Makes Business Sense				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Carnegie Mellon University ,Software Engineering Institute (SEI),Pittsburgh,PA,15213				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES Australian Software Engineering Conference (ASWEC 2006), 18-21 April, Sydney, Australia.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 74	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

BUSINESS SUCCESS REQUIRES SOFTWARE PROWESS



Software pervades every sector.

Software has become the bottom line for many organizations who never envisioned themselves in the software business.

© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 2

UNIVERSAL NEEDS

- Deploy new products (services) at a rapid pace
- Accommodate a growing demand for new product features across a wide spectrum of feature categories
- Connect products in increasingly unprecedented ways
- Exploit a rapidly changing technology base
- Gain a competitive edge



UNIVERSAL BUSINESS GOALS

HIGH QUALITY
QUICK TIME TO MARKET
EFFECTIVE USE OF
LIMITED RESOURCES
PRODUCT ALIGNMENT
LOW COST PRODUCTION
LOW COST MAINTENANCE
MASS CUSTOMIZATION
MIND SHARE

IMPROVED
EFFICIENCY
&
PRODUCTIVITY

© 2006 Carnegie Mellon University



THE ULTIMATE UNIVERSAL GOAL



© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 5

SOFTWARE (SYSTEM) STRATEGIES

PROCESS IMPROVEMENT

TECHNOLOGY INNOVATION

REUSE



© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 6

FEW SYSTEMS ARE UNIQUE

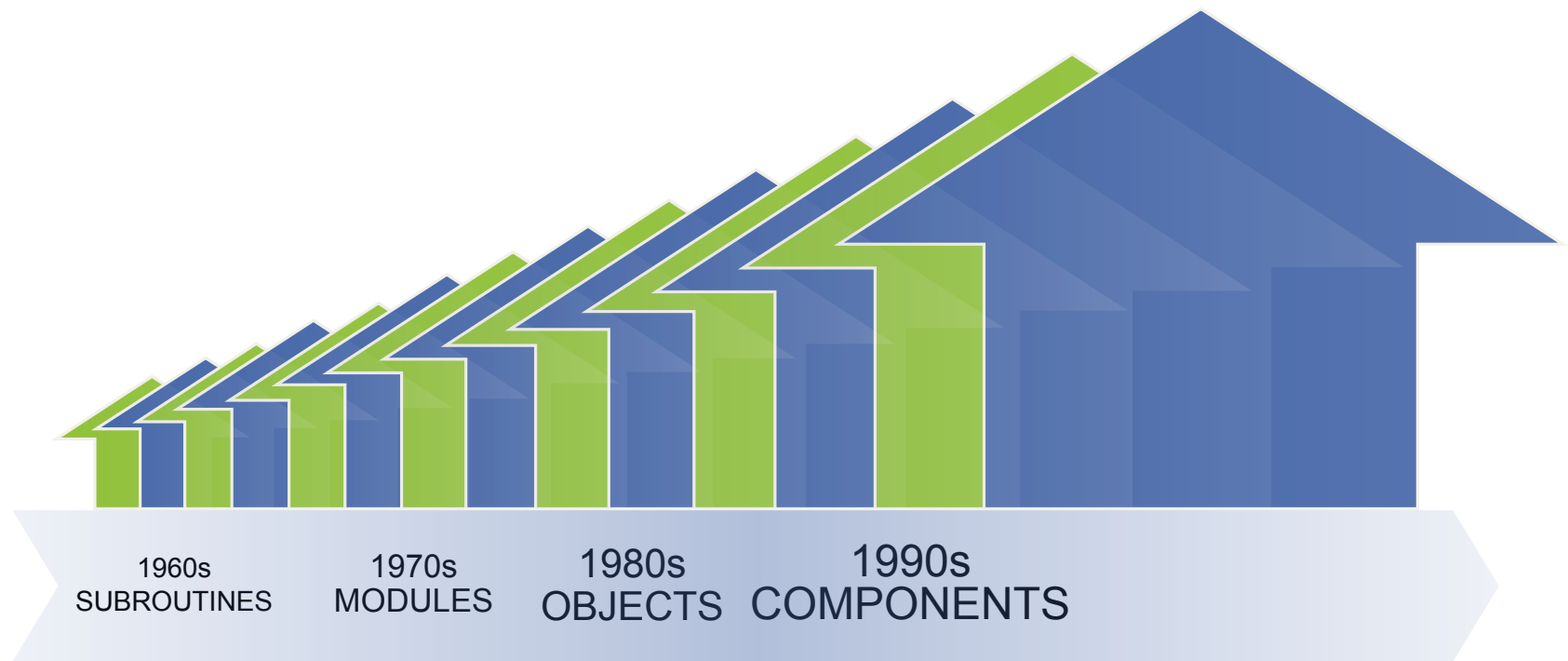


Most organizations produce families of similar systems, differentiated by features.

© 2006 Carnegie Mellon University



REUSE HISTORY



*Focus was small-grained and opportunistic.
Results fell short of expectations.*

© 2006 Carnegie Mellon University



BUT WHAT IS REUSE?

REUSE MEANS TAKING SOMETHING DEVELOPED FOR ONE SYSTEM AND USING IT IN ANOTHER.

“The Army XYZ System is built with 80% reuse.”

A statement like this is vacuous.

- It is not clear what is being reused.
- It is not clear that the “reuse” has any benefit.

Reusing code or components without an architecture focus and without pre-planning results in

- short-term perceived win
- long-term costs and problems



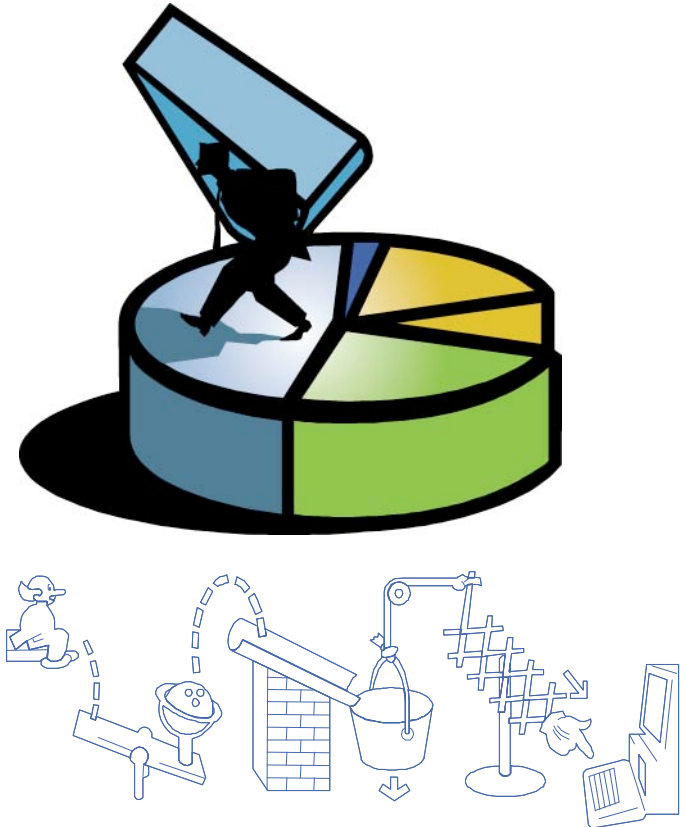
SOFTWARE REUSE FACT AND FICTION

THE FICTION:

“... and then we’ll be able to construct software systems by picking out parts and plugging them together, just like Tinkertoys ...”

THE FACT:

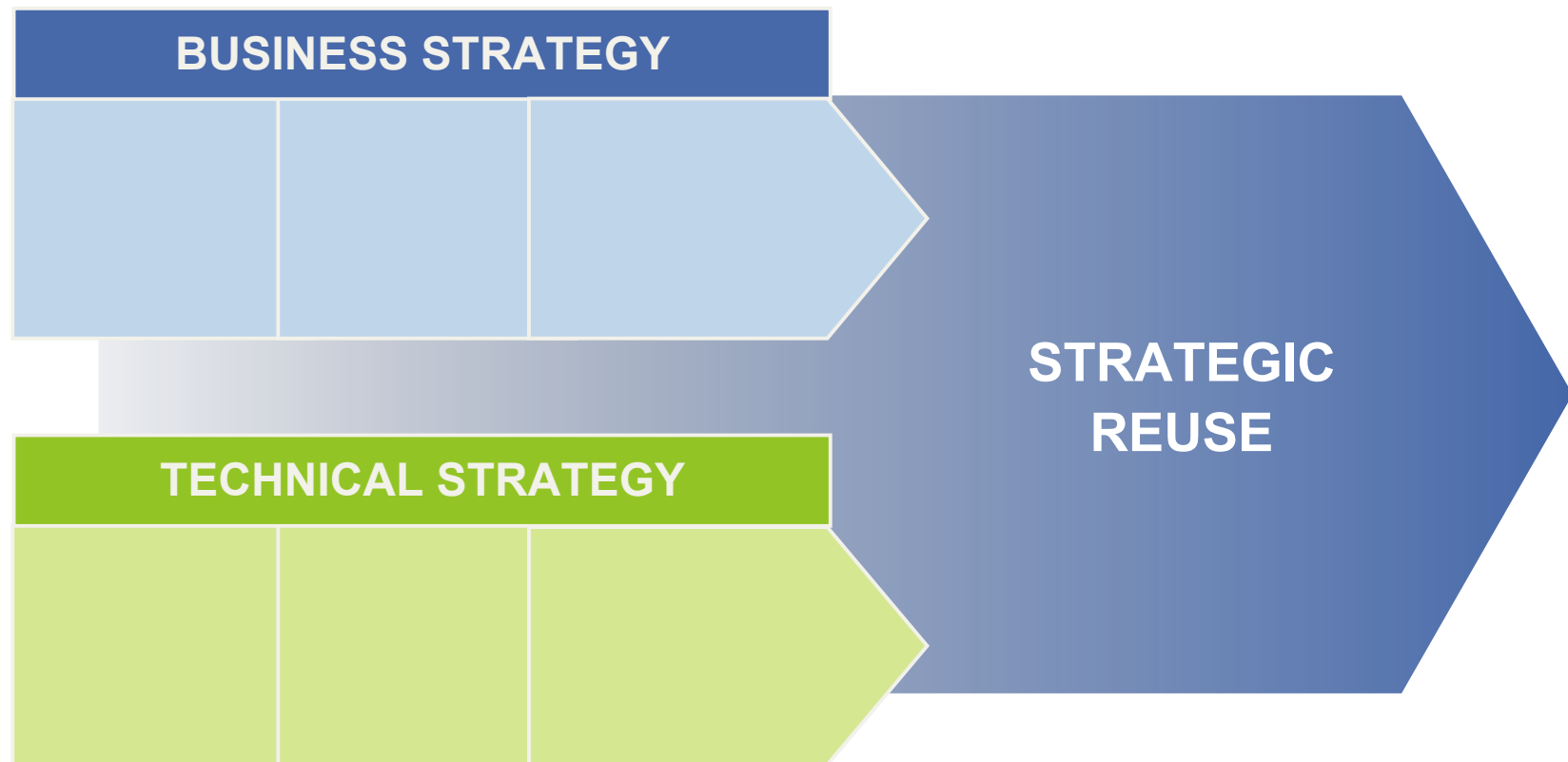
“It’s more like having a bathtub full of Tinkertoys, Legos, Erector Set parts, Lincoln Logs, Block City, and six other incompatible kits -- picking out parts that fit specific functions and expecting them to fit together.”



© 2006 Carnegie Mellon University



IMAGINE STRATEGIC REUSE



© 2006 Carnegie Mellon University



CELSIUSTECH: SHIP SYSTEM 2000

A FAMILY OF 55 SHIP SYSTEMS

- Integration test of 1-1.5 million
- SLOC requires 1-2 people.
- Rehosting to a new platform/OS takes 3 months.
- Cost and schedule targets are predictably met.
- Performance/distribution behavior are known in advance.
- Customer satisfaction is high.
- Hardware-to-software cost ratio changed from 35:65 to 80:20.



CUMMINS INC.: DIESEL CONTROL SYSTEMS

OVER 20 PRODUCT GROUPS WITH OVER 1,000 SEPARATE ENGINE APPLICATIONS

- Product cycle time was slashed from 250 person-months to a few person-months.
- Build and integration time was reduced from one year to one week.
- Quality goals are exceeded.
- Customer satisfaction is high.
- Product schedules are met.



© 2006 Carnegie Mellon University



NATIONAL RECONNAISSANCE OFFICE/ RAYTHEON: CONTROL CHANNEL TOOLKIT

GROUND-BASED SPACECRAFT COMMAND AND CONTROL SYSTEMS

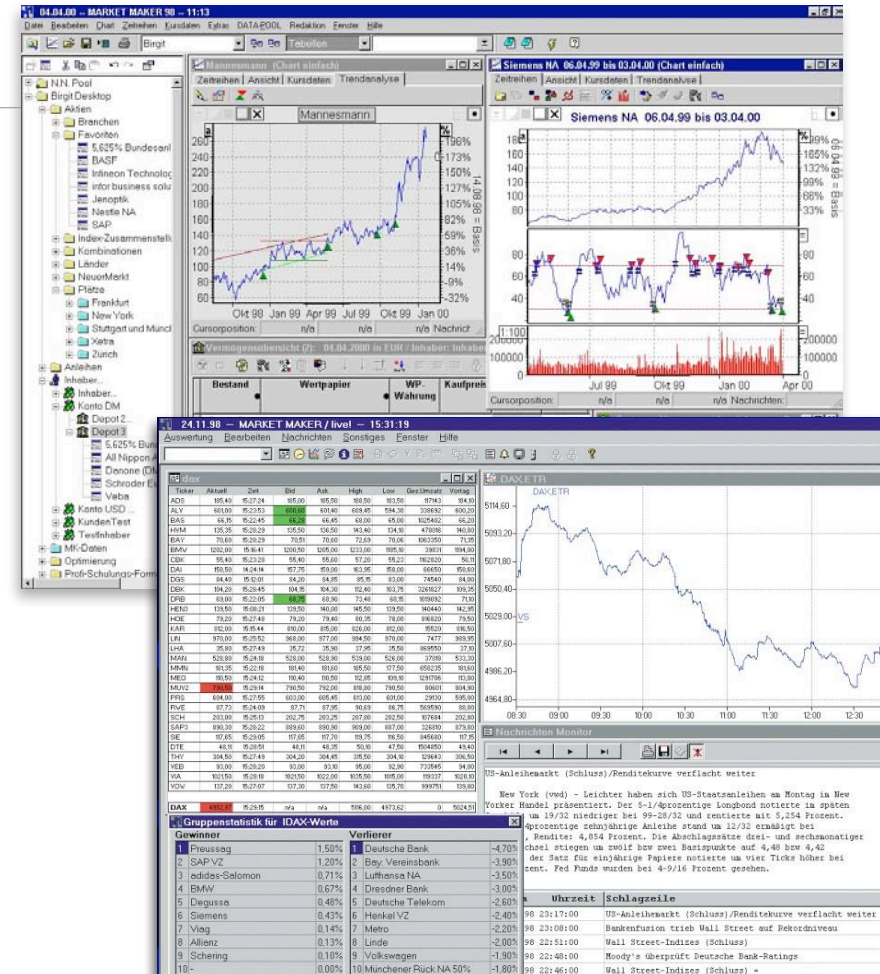
- increased quality by 10X
- incremental build time reduced from months to weeks
- software productivity increased by 7X
- development time and costs decreased by 50%
- decreased product risk



MARKET MAKER GMBH: MERGER

INTERNET-BASED STOCK MARKET SOFTWARE

- Each product is “uniquely” configured.
- Putting up a customized system takes three days.



© 2006 Carnegie Mellon University



NOKIA MOBILE PHONES

PRODUCT LINES WITH 25-30 NEW PRODUCTS PER YEAR

Across products there are

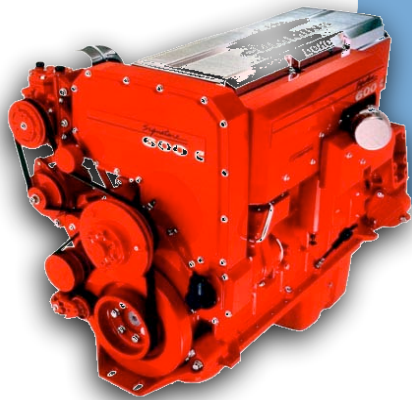
- varying number of keys
- varying display sizes
- varying sets of features
- 58 languages supported
- 130 countries served
- multiple protocols
- needs for backwards compatibility
- configurable features
- needs for product behavior
- change after release



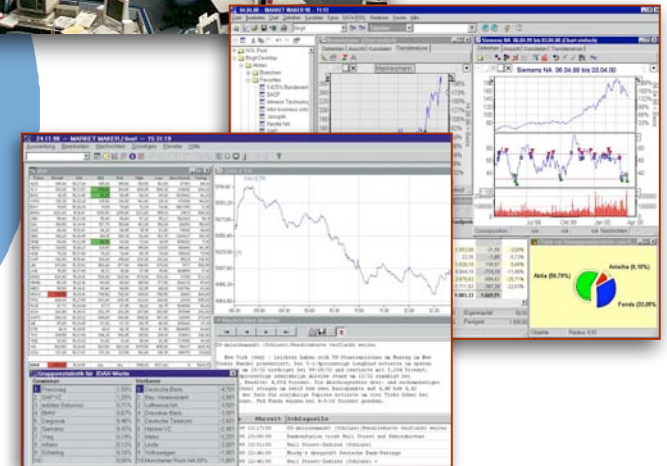
© 2006 Carnegie Mellon University



HOW DID THEY DO IT?



SOFTWARE
PRODUCT
LINES



© 2006 Carnegie Mellon University



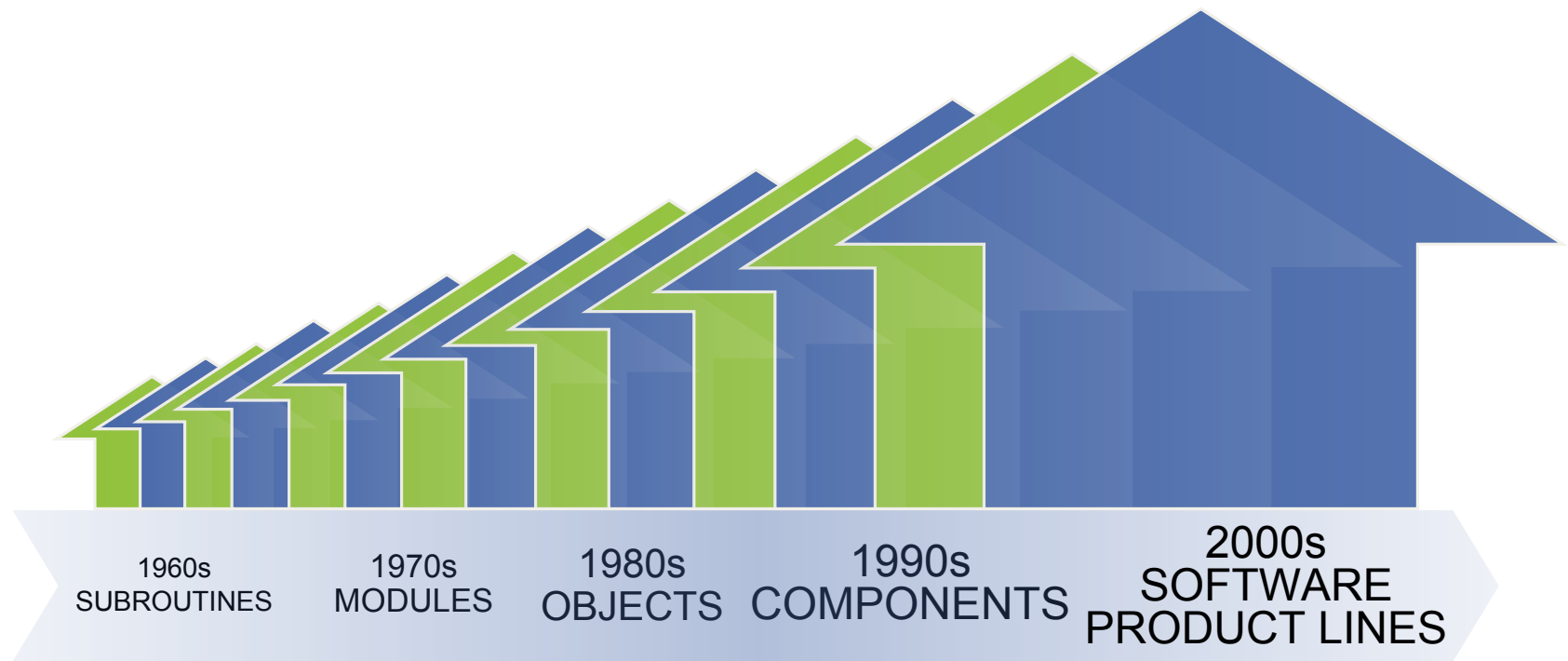
Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 17

REUSE HISTORY: FROM AD HOC TO SYSTEMATIC



© 2006 Carnegie Mellon University



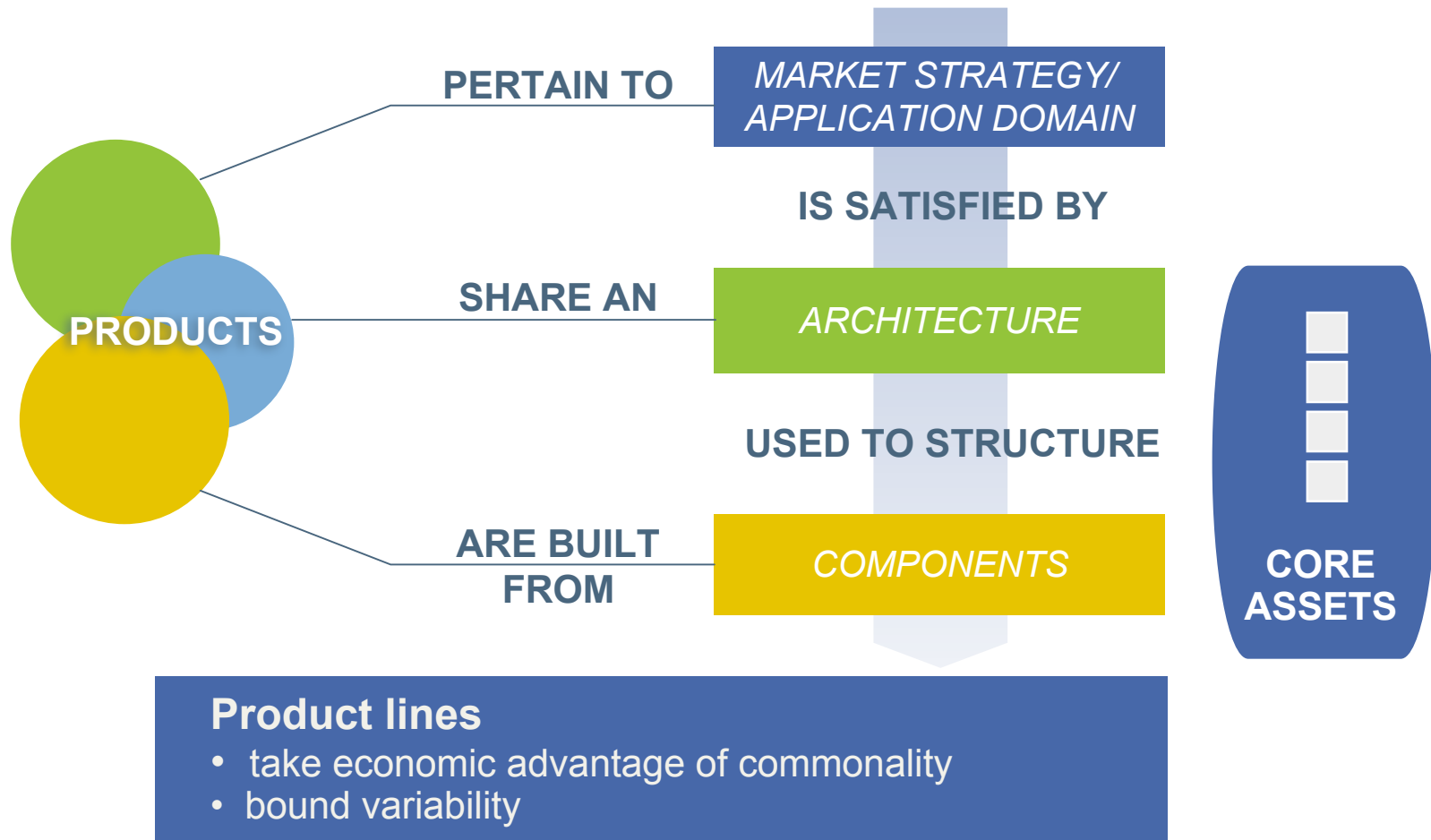
WHAT IS A SOFTWARE PRODUCT LINE?

A software product line is a **set** of software-intensive systems sharing a **common, managed set of features** that satisfy the specific needs of a **particular market segment or mission** and that are **developed from a common set of core assets** in a **prescribed way**.

© 2006 Carnegie Mellon University



SOFTWARE PRODUCT LINES



© 2006 Carnegie Mellon University



HOW DO PRODUCT LINES HELP?

PRODUCT LINES AMORTIZE THE INVESTMENT IN THESE AND OTHER *CORE ASSETS*:

- requirements and requirements analysis
- domain model
- software architecture and design
- performance engineering
- documentation
- test plans, test cases, and test data
- people: their knowledge and skills
- processes, methods, and tools
- budgets, schedules, and work plans
- Components

PRODUCT LINES = STRATEGIC REUSE

EARLIER
LIFE CYCLE
REUSE



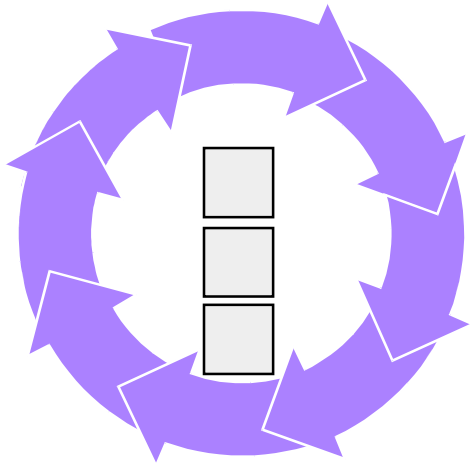
MORE
BENEFIT

© 2006 Carnegie Mellon University

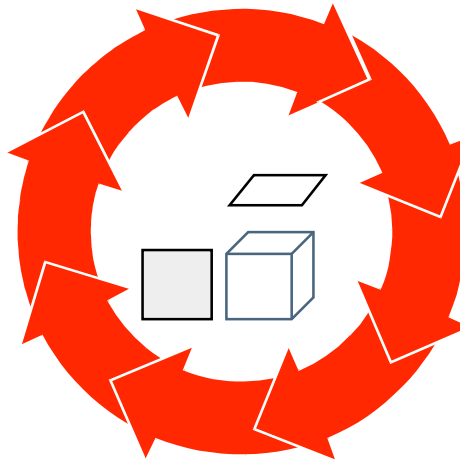


THE KEY CONCEPTS

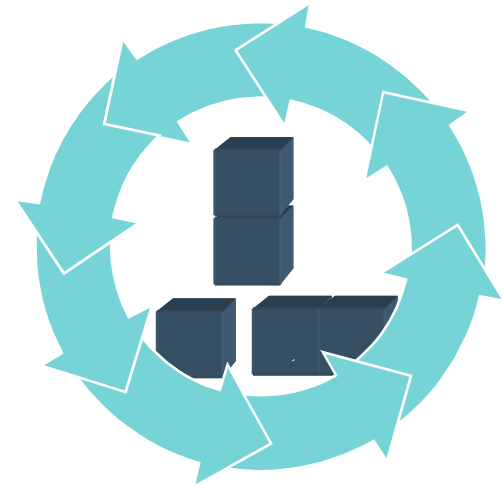
Use of a core
asset base



in production



of a related
set of products

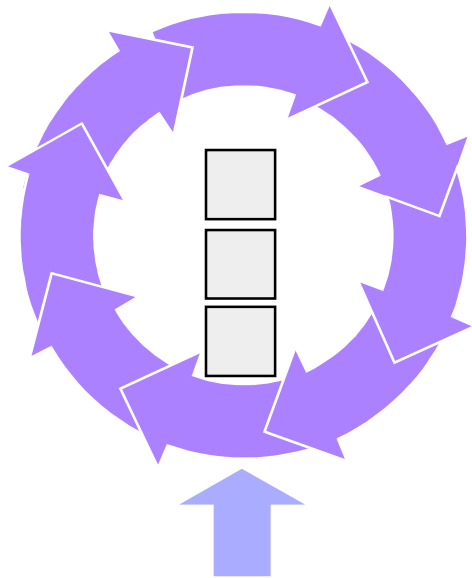


© 2006 Carnegie Mellon University



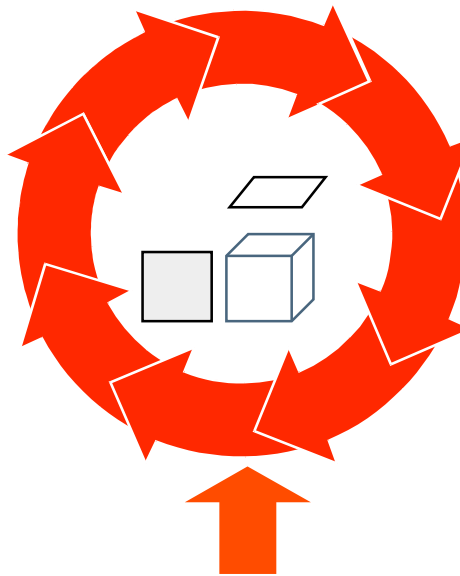
THE KEY CONCEPTS

Use of a core
asset base



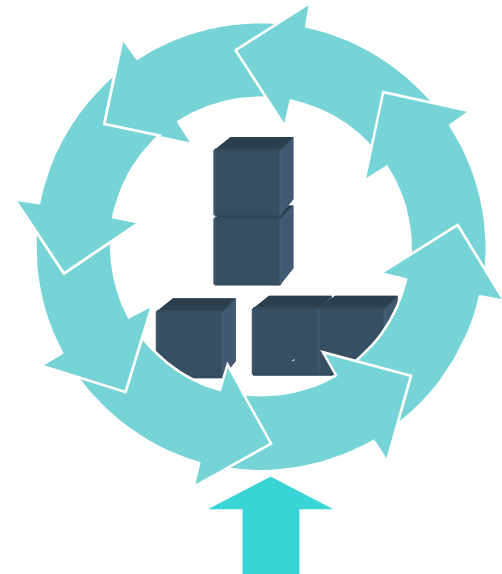
Architecture

in production



Production Plan

of a related
set of products



Scope Definition
Business Case

© 2006 Carnegie Mellon University



SOFTWARE PRODUCT LINES ARE NOT

FORTUITOUS SMALL-GRAINED REUSE

- reuse libraries containing algorithms, modules, objects, or components

SINGLE-SYSTEM DEVELOPMENT WITH REUSE

- borrowing opportunistically from previous efforts
- modifying code as necessary for the single system only

JUST COMPONENT-BASED DEVELOPMENT

- selecting components from an in-house library or the marketplace with no architecture focus

JUST A CONFIGURABLE ARCHITECTURE

- a good start, but only part of the reuse potential

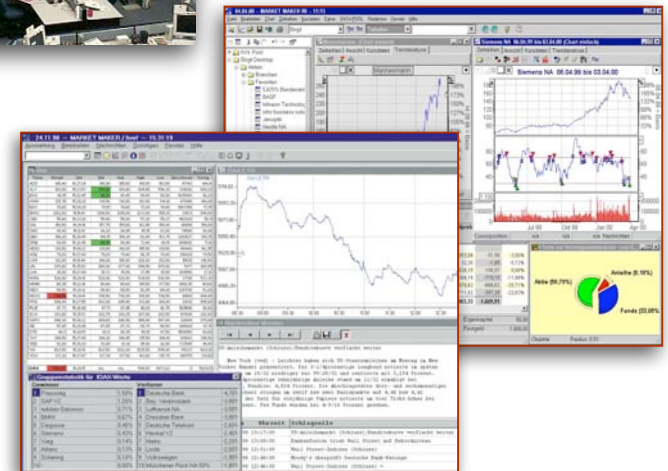
JUST A SET OF TECHNICAL STANDARDS

- constraining choices without an architecture-based reuse strategy

© 2006 Carnegie Mellon University



PRODUCT LINES ARE



Software product lines involve strategic, planned reuse that yields predictable results.

© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 25

COMMERCIAL EXAMPLES

SUCCESSFUL SOFTWARE PRODUCT LINES HAVE BEEN BUILT FOR FAMILIES OF

- mobile phones
- command and control ship systems
- ground-based spacecraft systems
- avionics systems
- command and control/situation awareness systems
- pagers
- engine control systems
- billing systems
- web-based retail systems
- printers
- consumer electronic products
- acquisition management enterprise systems
- financial and tax systems
- medical devices

© 2006 Carnegie Mellon University



REAL WORLD MOTIVATION

ORGANIZATIONS USE PRODUCT LINE PRACTICES TO:

- achieve large scale productivity gains
- improve time to market
- maintain market presence
- sustain unprecedented growth
- compensate for an inability to hire
- achieve systematic reuse goals
- improve product quality
- increase customer satisfaction
- enable mass customization
- get control of diverse product configurations
- achieve greater market agility



© 2006 Carnegie Mellon University



SUMMARY: ORGANIZATIONAL BENEFITS

IMPROVED PRODUCTIVITY

- by as much as 10x

DECREASED TIME TO MARKET (TO FIELD, TO LAUNCH...)

- by as much as 10x

DECREASED COST

- by as much as 60%

DECREASED LABOR NEEDS

- by as much as 10X fewer software developers

INCREASED QUALITY

- by as much as 10X fewer defects

*Product line practice permits **predictable** “faster, better, cheaper.”*

© 2006 Carnegie Mellon University



COSTS OF A SOFTWARE PRODUCT LINE

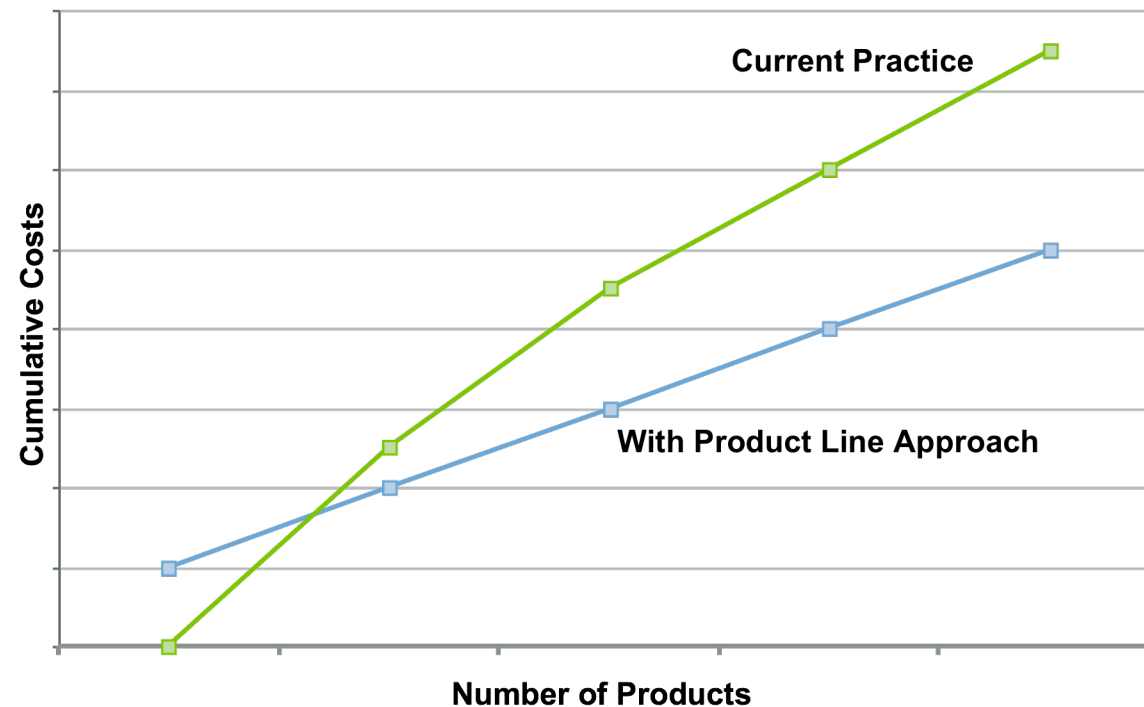
Core Assets	Costs
Architecture	Must support variation inherent in the product line
Software Components	Must be designed to be general without a loss of performance; must build in support for variation points
Test Plans, Test Cases, Test Data	Must consider variation points and multiple instances of the product line
Business Case and Market Analysis	Must address a family of software products, not just one product
Project Plans	Must be generic or be made extensible to accommodate product variations
Tools and Processes	Must be more robust
People, Skills, Training	Must involve training and expertise centered around the assets and procedures associated with the product line



© 2006 Carnegie Mellon University



ECONOMICS OF PRODUCT LINES

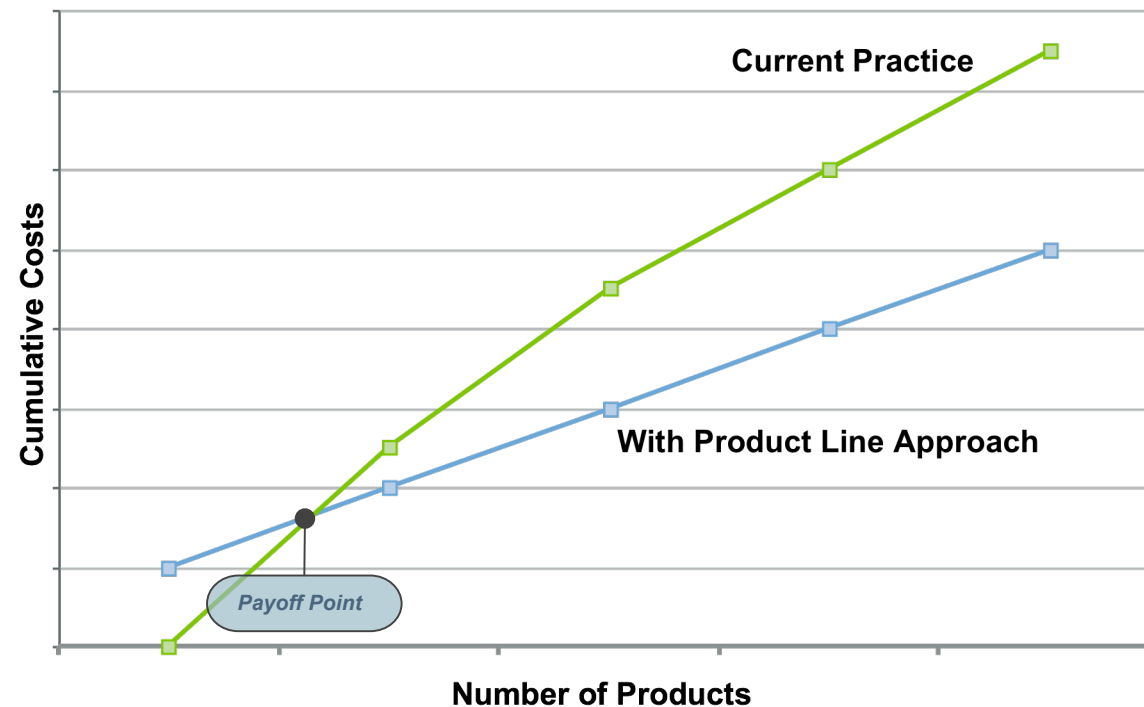


Weiss, D.M. & Lai, C.T.R..
Software Product-Line Engineering: A Family-Based Software Development Process
Reading, MA: Addison-Wesley, 1999.

© 2006 Carnegie Mellon University



ECONOMICS OF PRODUCT LINES

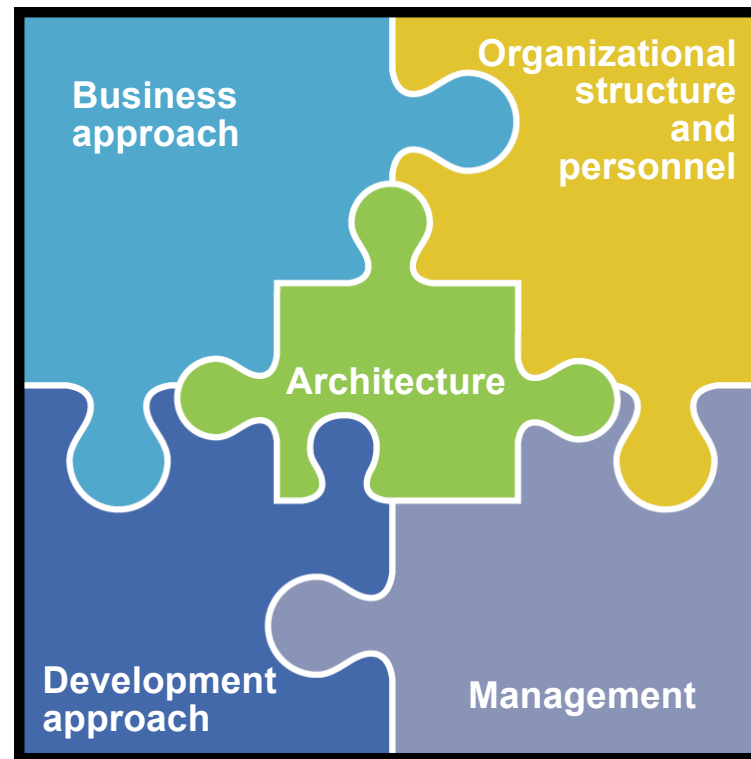


Weiss, D.M. & Lai, C.T.R..
Software Product-Line Engineering: A Family-Based Software Development Process
Reading, MA: Addison-Wesley, 1999.

© 2006 Carnegie Mellon University



NECESSARY CHANGES



The product line architecture is the foundation of everything.

© 2006 Carnegie Mellon University



WHY IS SOFTWARE ARCHITECTURE IMPORTANT?

Represents *earliest* design decisions



- hardest to change
- most critical to get right
- communication vehicle among stakeholders

First design artifact addressing



- performance
- modifiability
- reliability
- security

Key to systematic *reuse*



- transferable, reusable abstraction

The **right architecture** paves the way for system **success**.
The **wrong architecture** usually spells some form of **disaster**.

© 2006 Carnegie Mellon University



PRODUCT LINE PRACTICE

CONTEXTS FOR PRODUCT LINES VARY WIDELY, BASED ON

- nature of products
- nature of market or mission
- business goals
- organizational infrastructure
- workforce distribution
- process discipline
- artifact maturity

**But there are
universal essential
activities and practices.**

© 2006 Carnegie Mellon University



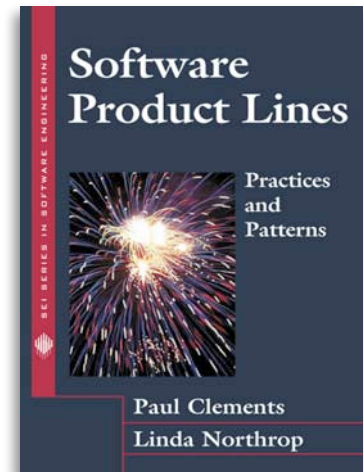
THE SEI FRAMEWORK FOR SOFTWARE PRODUCT LINE PRACTICESM

The SEI Framework for Software Product Line Practice is a conceptual framework that describes the essential activities and twenty-nine practice areas necessary for successful software product lines.

The Framework, originally conceived in 1998, is evolving based on the experience and information provided by the community.

Version 4.0 – in *Software Product Lines: Practices and Patterns*

Version 4.2 – <http://www.sei.cmu.edu/productlines/framework.html>



© 2006 Carnegie Mellon University



SEI INFORMATION SOURCES

Case studies, experience reports, and surveys

Workshops and conferences



Applied research

Collaborations with customers on actual product lines

© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 36

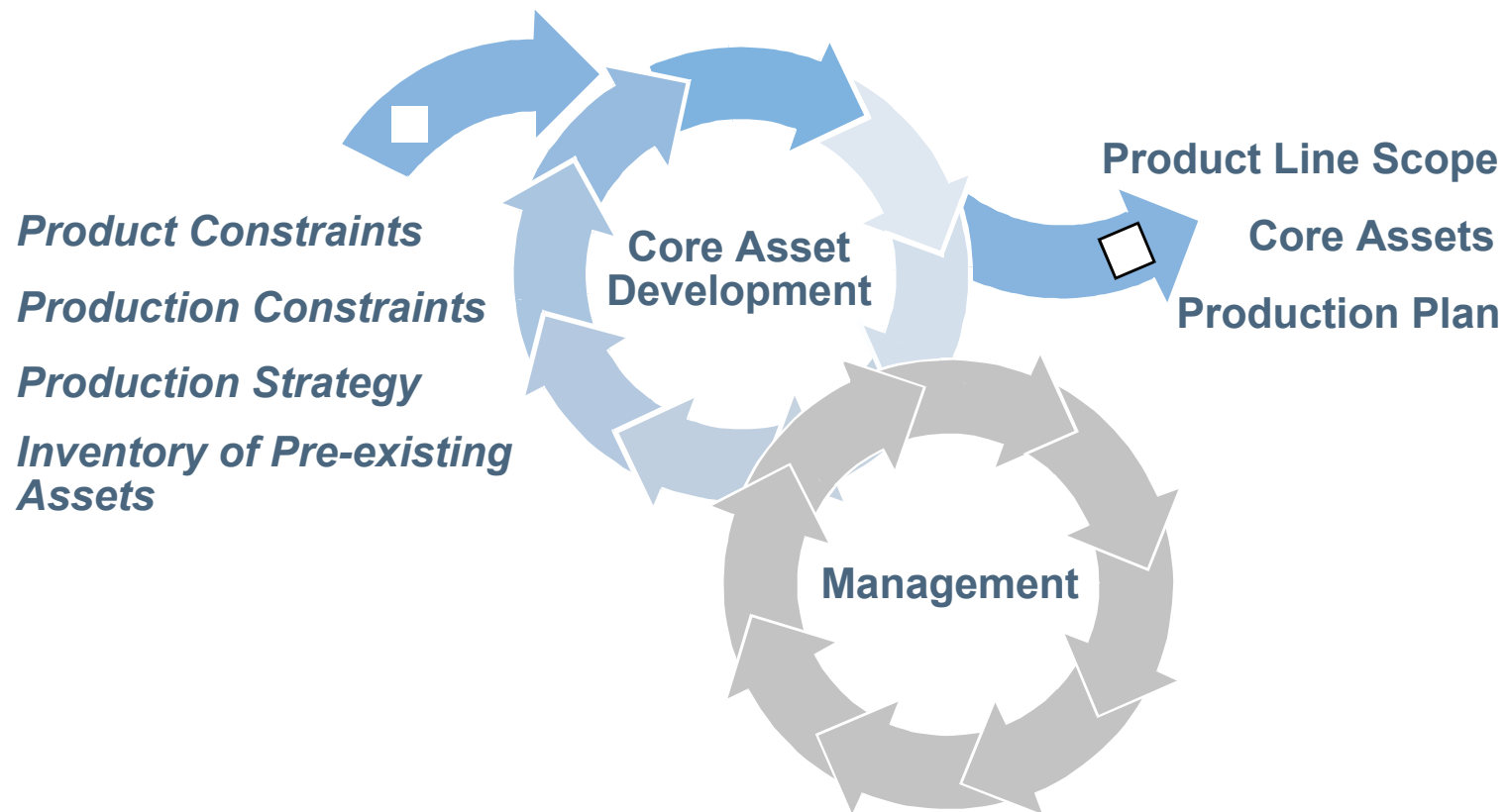
THE THREE ESSENTIAL ACTIVITIES



© 2006 Carnegie Mellon University



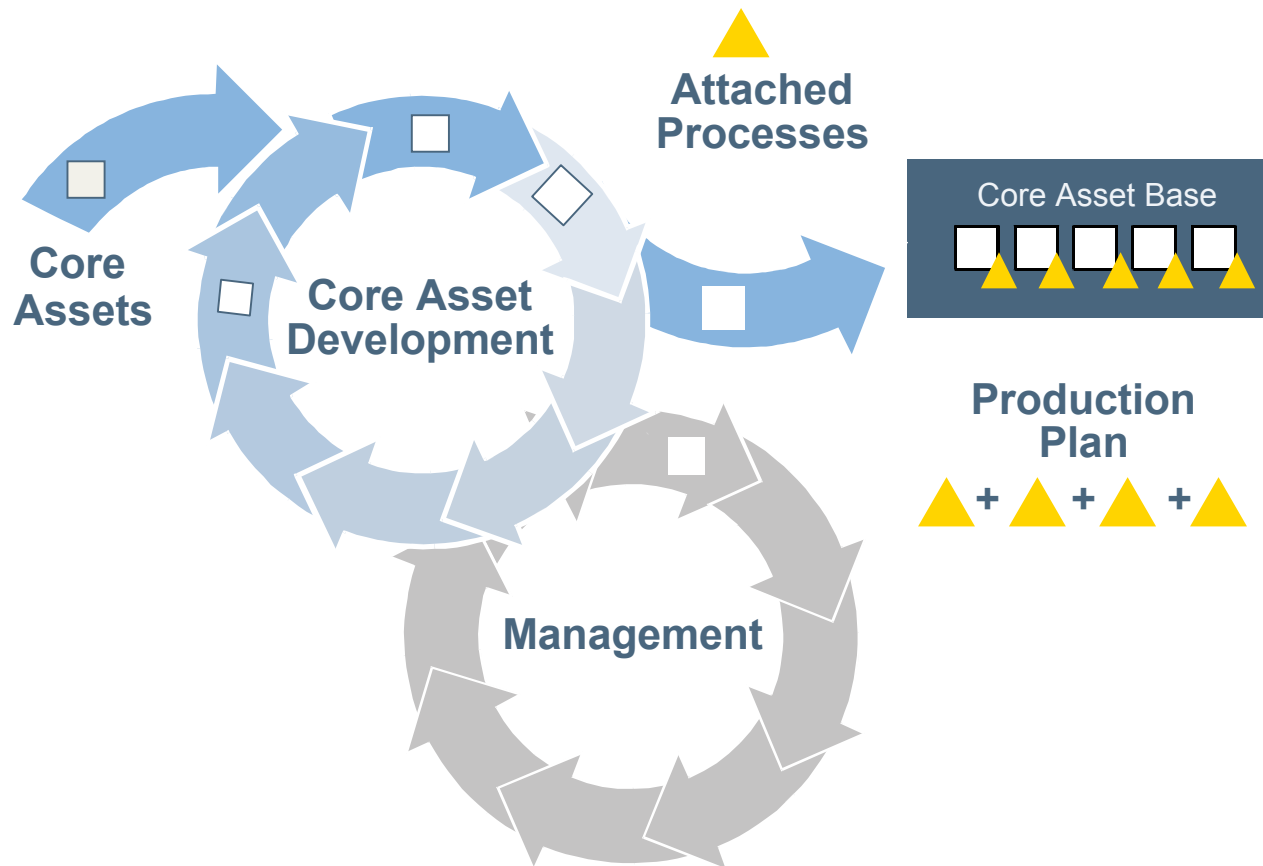
CORE ASSET DEVELOPMENT



© 2006 Carnegie Mellon University



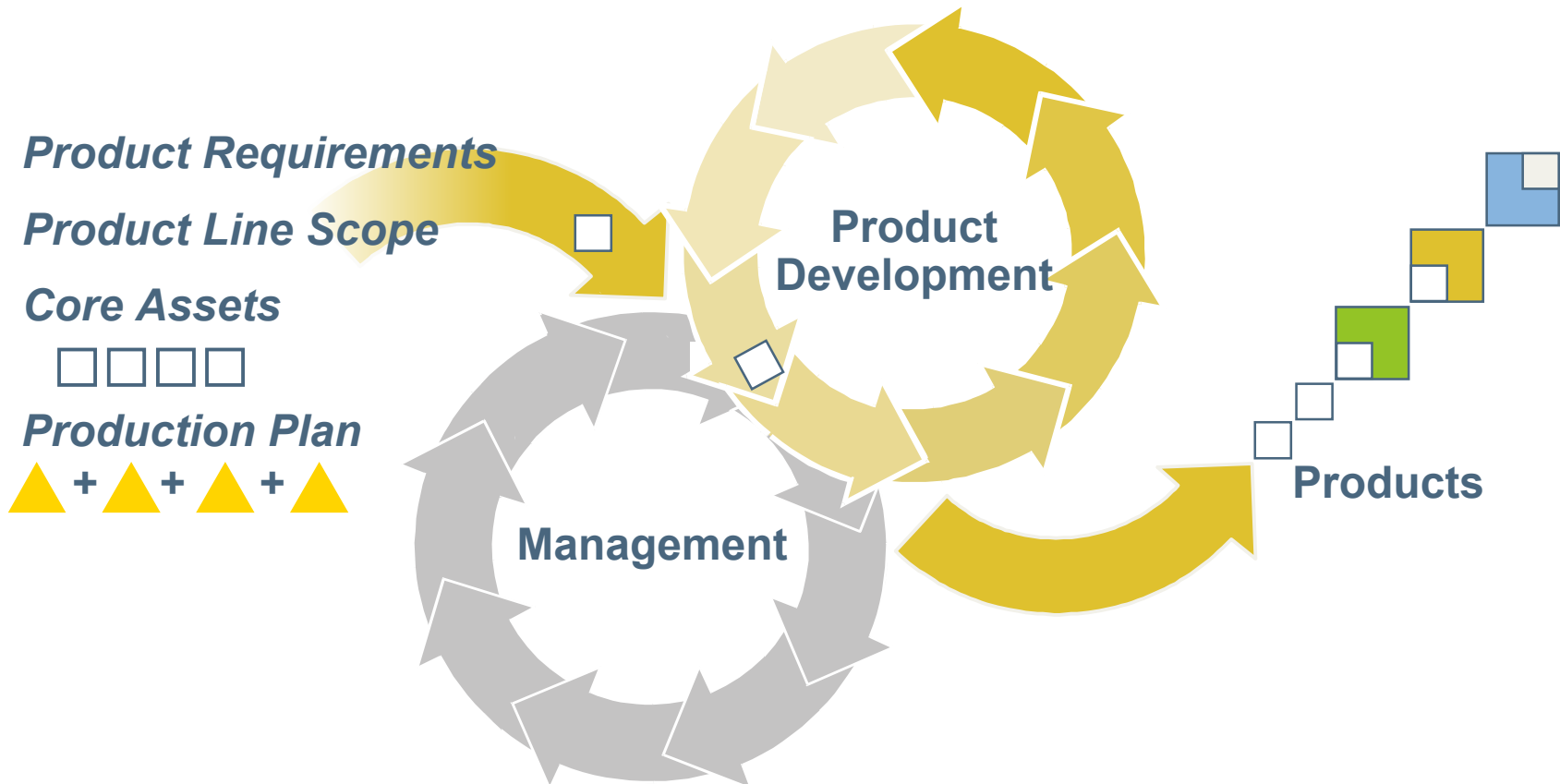
ATTACHED PROCESSES



© 2006 Carnegie Mellon University



PRODUCT DEVELOPMENT



© 2006 Carnegie Mellon University



PRODUCT DEVELOPMENT



© 2006 Carnegie Mellon University



MANAGEMENT



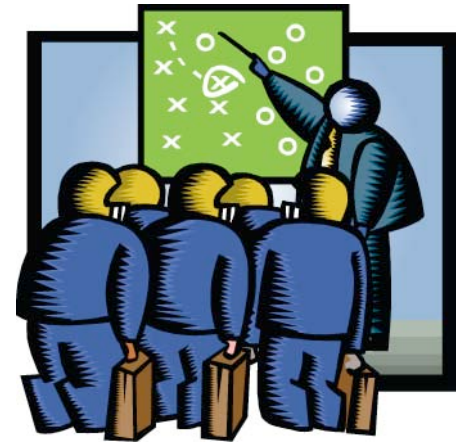
© 2006 Carnegie Mellon University



MANAGEMENT

MANAGEMENT AT MULTIPLE LEVELS PLAYS A CRITICAL ROLE IN THE SUCCESSFUL PRODUCT LINE PRACTICE BY

- achieving the right organizational structure
- allocating resource
- coordinating and supervising
- providing training
- rewarding employees appropriately
- developing and communicating an acquisition strategy
- managing external interfaces
- creating and implementing a product line adoption plan
- launching and institutionalizing the approach in a manner appropriate to the organization



© 2006 Carnegie Mellon University



MANAGING A SOFTWARE PRODUCT LINE REQUIRES LEADERSHIP

A KEY ROLE FOR A SOFTWARE PRODUCT LINE MANAGER IS THAT OF CHAMPION.

The champion must

- set and maintain the vision
- ensure that the appropriate goals and measures are in place
- “sell” the product line up and down the chain
- sustain morale
- deflect potential derailments
- solicit feedback and continuously improve the approach

© 2006 Carnegie Mellon University



ESSENTIAL PRODUCT LINE ACTIVITIES



*Each of these is essential,
as is the blending of all three.*

© 2006 Carnegie Mellon University



DIFFERENT APPROACHES - 1

PROACTIVE: DEVELOP THE CORE ASSETS FIRST.

- Develop the scope first and use it as a “mission” statement.
- Products come to market quickly with minimum code writing.
- requires upfront investment and predictive knowledge

REACTIVE: START WITH ONE OR MORE PRODUCTS.

- From them, generate the product line core assets and then future products; the scope evolves more dramatically.
- much lower cost of entry
- The architecture and other core assets must be robust, extensible, and appropriate to future product line needs.

© 2006 Carnegie Mellon University



DIFFERENT APPROACHES - 2

INCREMENTAL:

In either a reactive or proactive approach, it is possible to develop the core asset base in stages, while planning from the beginning to develop a product line.

- Develop part of the core asset base, including the architecture and some of the components.
- Develop one or more products.
- Develop part of the rest of the core asset base.
- Develop more products.
- Evolve more of the core asset base.
- ...



ALTERNATE TERMINOLOGY

OUR TERMINOLOGY		ALTERNATE TERMINOLOGY
Product Line		Product Family
Core Assets		Platform
Business Unit		Product Line
Product		Customization
Core Asset Development		Domain Engineering
Product Development		Application Engineering

© 2006 Carnegie Mellon University



DRIVING THE ESSENTIAL ACTIVITIES

BENEATH THE LEVEL OF THE ESSENTIAL ACTIVITIES ARE ESSENTIAL PRACTICES THAT FALL INTO PRACTICE AREAS.

A **practice area** is a body of work or a collection of activities that an organization must master to successfully carry out the essential work of a product line.



PRACTICE AREAS CATEGORIES



© 2006 Carnegie Mellon University



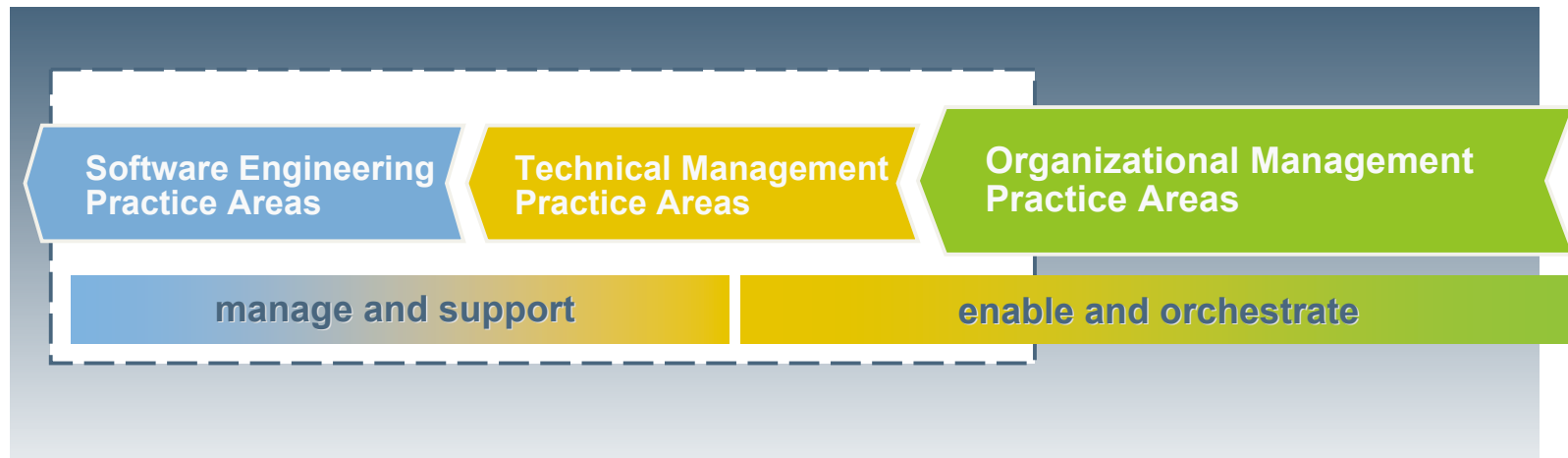
Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 50

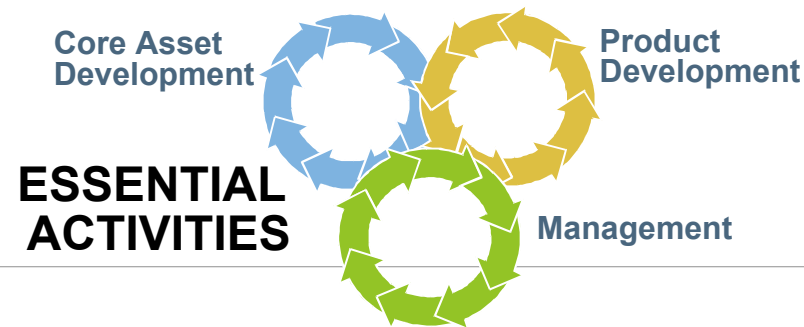
RELATIONSHIPS AMONG CATEGORIES OF PRACTICE AREAS



© 2006 Carnegie Mellon University



FRAMEWORK



PRACTICE AREAS		
Software Engineering	Technical Management	Organizational Management
Architecture Definition	Configuration Management	Building a Business Case
Architecture Evaluation	Data Collection, Metrics, and Tracking	Customer Interface Management
Component Development	Make/Buy/Mine/Commission Analysis	Developing an Acquisition Strategy
COTS Utilization	Process Definition	Funding
Mining Existing Assets	Scoping	Launching and Institutionalizing
Requirements Engineering	Technical Planning	Market Analysis
Software System Integration	Technical Risk Management	Operations
Testing	Tool Support	Organizational Planning
Understanding Relevant Domains		Organizational Risk Management
		Structuring the Organization
		Technology Forecasting
		Training

© 2006 Carnegie Mellon University



DILEMMA: HOW DO YOU APPLY THE 29 PRACTICE AREAS?

ORGANIZATIONS STILL HAVE TO FIGURE OUT HOW TO PUT THE PRACTICE AREAS INTO PLAY.

Twenty-nine is a big number.



HELP TO MAKE IT HAPPEN



PRACTICE AREAS

Software Engineering

Technical Management

Organizational Management

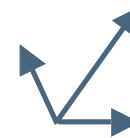
GUIDANCE



Case Studies



Patterns



Probe

© 2006 Carnegie Mellon University



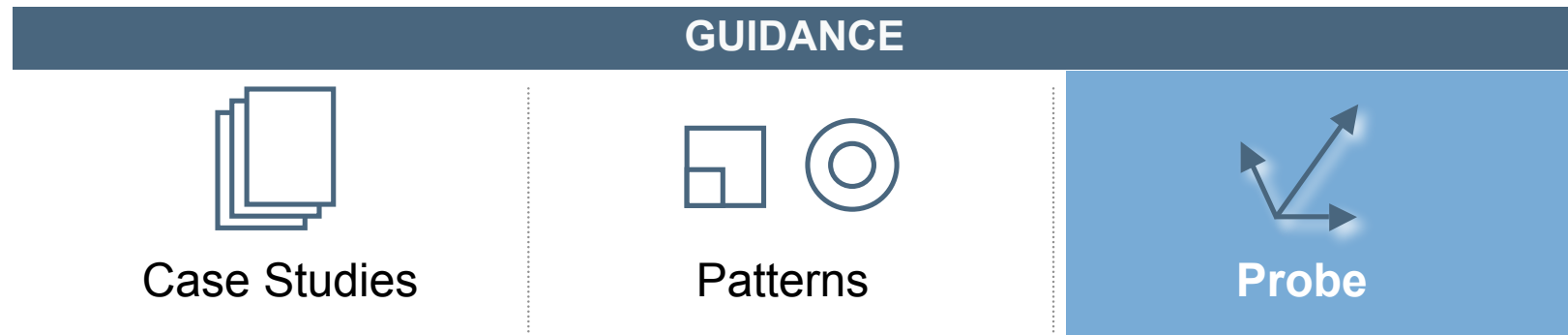
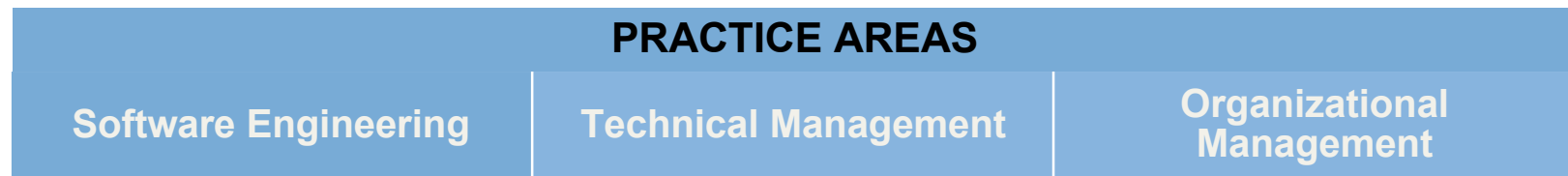
Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 54

HELP TO MAKE IT HAPPEN





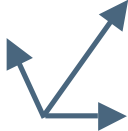
e Mellon Univer:



HELP TO MAKE IT HAPPEN



PRACTICE AREAS		
Software Engineering	Technical Management	Organizational Management

GUIDANCE		
 Case Studies	 Patterns	 Probe

© 2006 Carnegie Mellon University

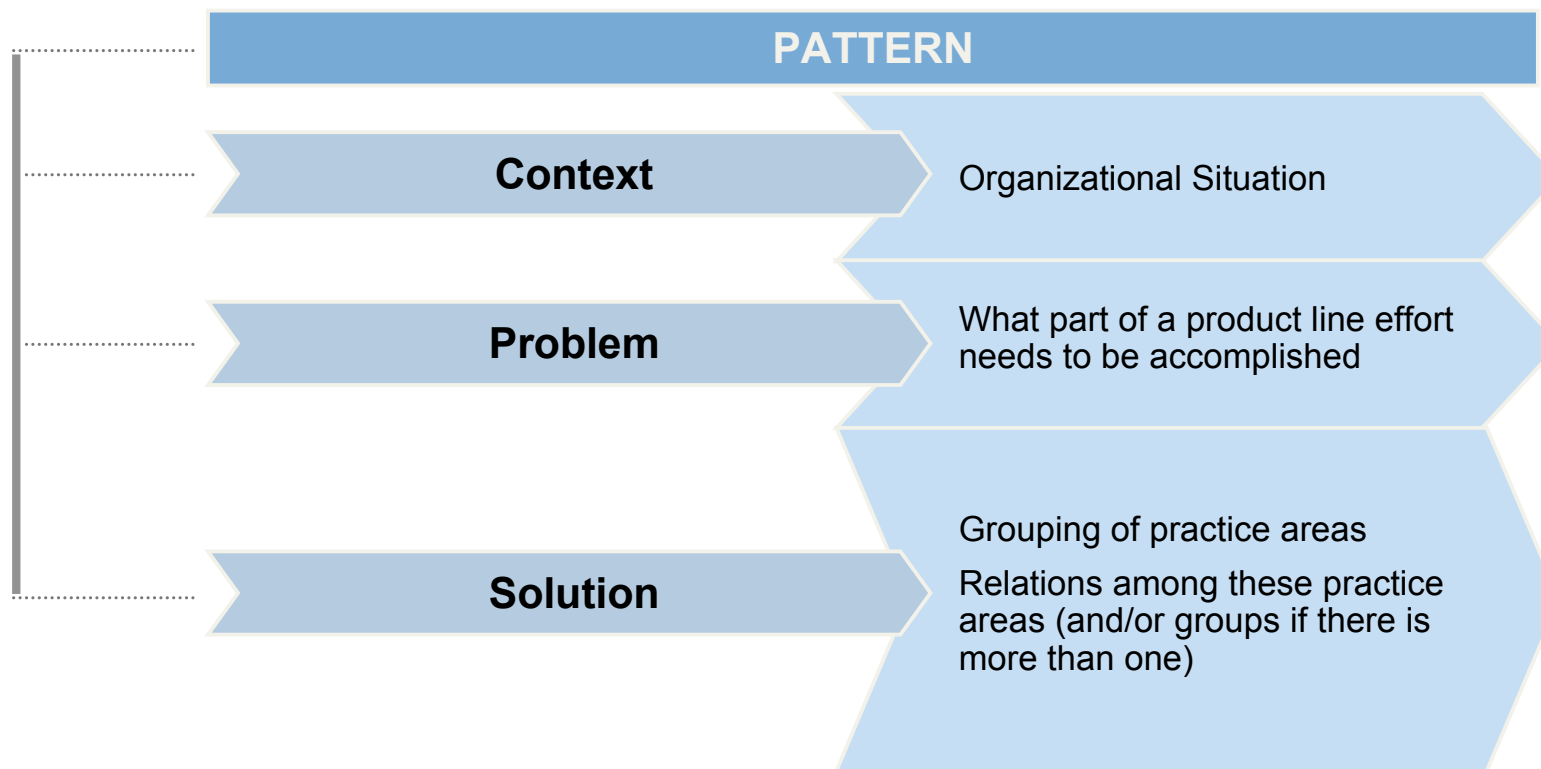


PATTERNS CAN HELP

- Patterns are a way of expressing common context and problem-solution pairs.
- Patterns have been found to be useful in building architecture, economics, software architecture, software design, software implementation, process improvement, and others.
- Patterns assist in effecting a divide and conquer approach.



SOFTWARE PRODUCT LINE PRACTICE PATTERN



© 2006 Carnegie Mellon University



WHAT TO BUILD PATTERN - 1

NAME:

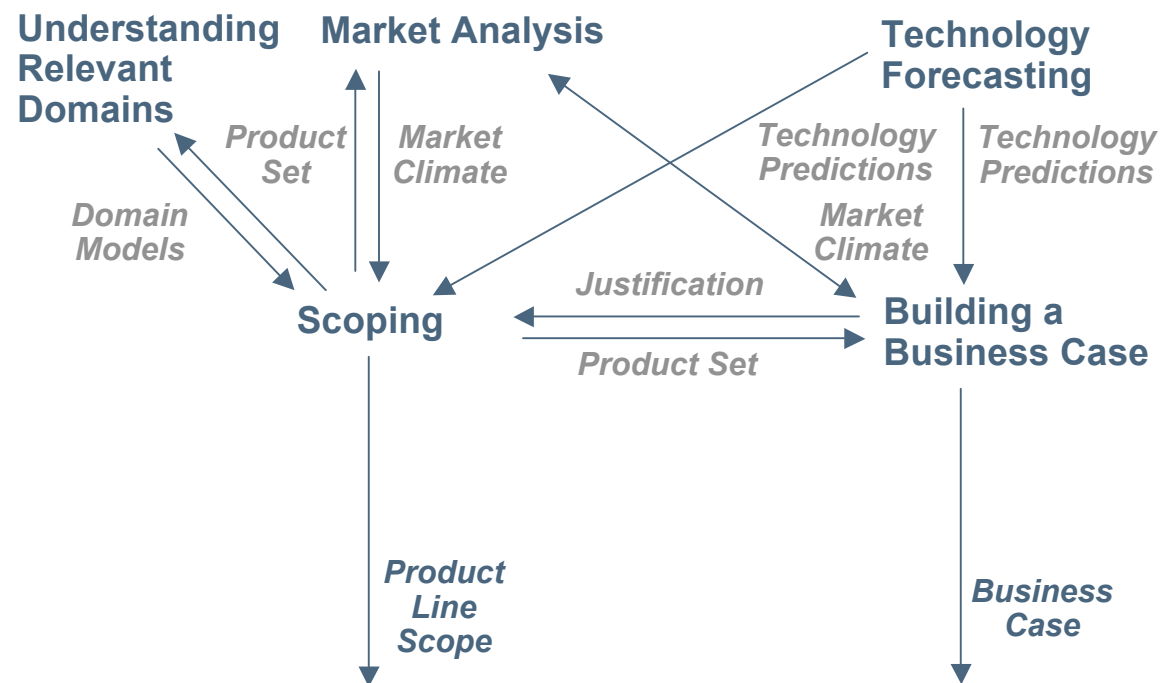
The ***What to Build*** pattern helps an organization determine what products ought to be in its software product line – what products to build.

CONTEXT:

An organization has decided to field a software product line and knows the general product area for the set of products.



WHAT TO BUILD PATTERN - 2



© 2006 Carnegie Mellon University



FACTORY PATTERN - 1

NAME:

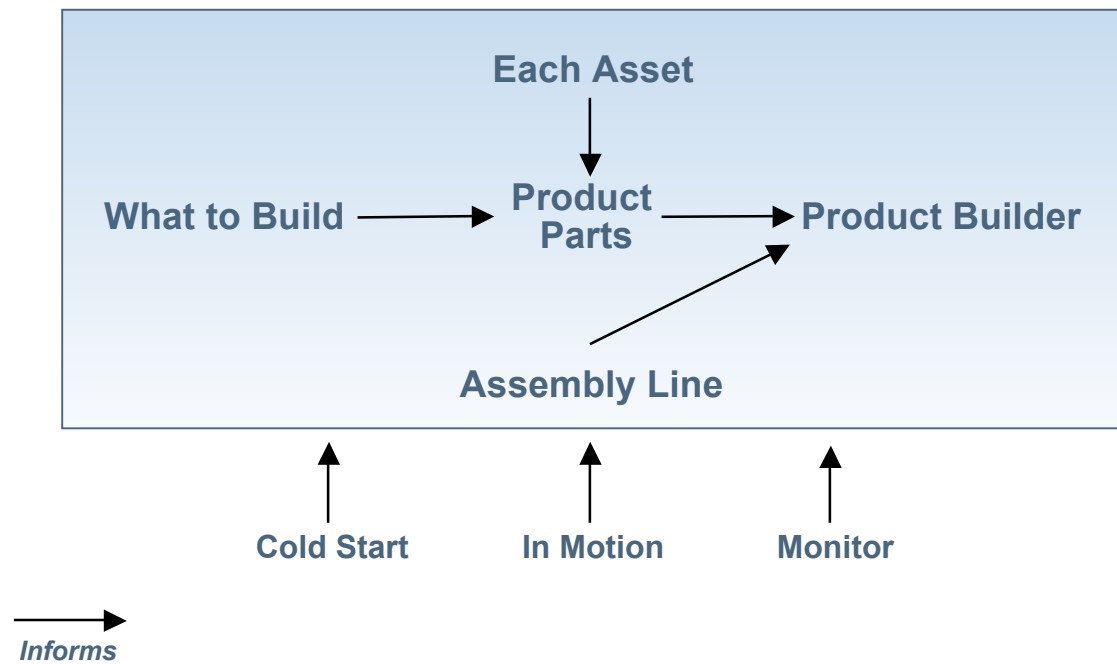
The **Factory** patterns is a composite pattern that describes the entire product line organization.

CONTEXT:

An organization is considering (or fielding) a product line.



FACTORY PATTERN - 2



Dynamic Structure

© 2006 Carnegie Mellon University



CURRENT SET OF PATTERNS

Pattern	Variants
Assembly Line	
Cold Start	Warm Start
Curriculum	
Each Asset	Each Asset Apprentice Evolve Each Asset
Essentials Coverage	
Factory	Adoption Factory
In Motion	
Monitor	
Process	Process Improvement
Product Parts	Green Field Barren Field Plowed Field
What to Build	Analysis Forced March

© 2006 Carnegie Mellon Univer



THE ADOPTION ENDGAME

EFFECTIVELY ACHIEVE AN *OPERATIONAL PRODUCT LINE*.

- have
 - a core asset base
 - supportive processes and organizational structures
- develop products from that asset base in a way that achieves business goals
- improve and extend the software product line adoption effort as long as it makes sense



BARRIERS TO PRODUCT LINE ADOPTION



Cost, cost,
&
cost....

You have to
invest to
eventually
SAVE.

© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 65

BARRIERS TO PRODUCT LINE ADOPTION



© 2006 Carnegie Mellon University



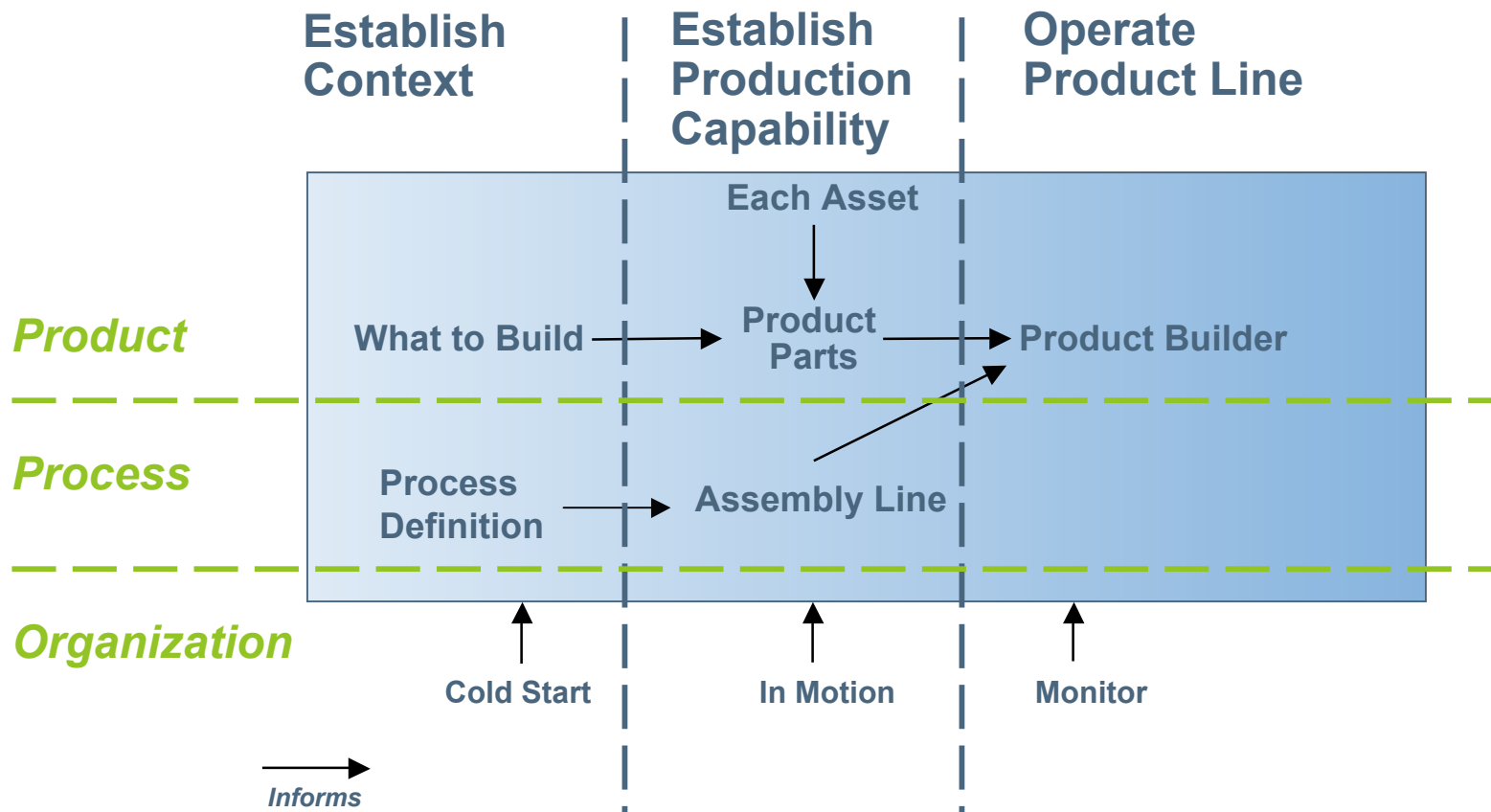
Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 66

THE ADOPTION FACTORY PATTERN



© 2006 Carnegie Mellon University



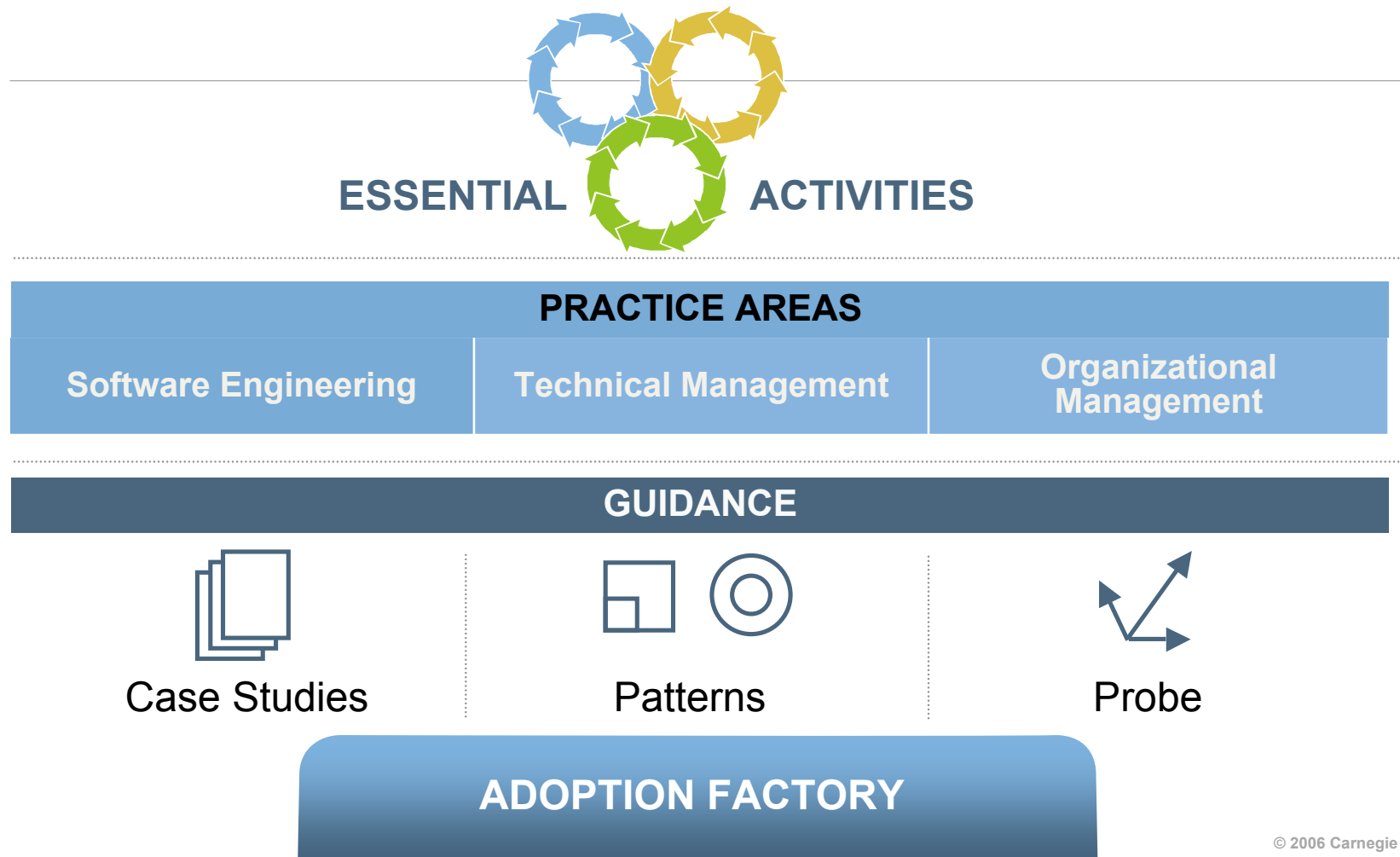
ASSOCIATED PRACTICE AREAS

	Establish Context	Establish Production Capability	Operate Product Line
Product	<ul style="list-style-type: none"> • Marketing Analysis • Understanding Relevant Domains • Technology Forecasting • Building a Business Case • Scoping 	<ul style="list-style-type: none"> • Requirements Engineering • Architecture Definition • Architecture Evaluation • Mining Existing Assets • Component Development • COTS Utilization • Software System Integration • Testing 	<ul style="list-style-type: none"> • Requirements Engineering • Architecture Definition • Architecture Evaluation • Mining Existing Assets • Component Development • COTS Utilization • Software System Integration • Testing
Process	<ul style="list-style-type: none"> • Process Definition 	<ul style="list-style-type: none"> • Make/Buy/Mine/Commission • Configuration Management • Tool Support • Data Collection, Metrics, Tracking • Technical Planning • Technical Risk Management 	
Organization	<ul style="list-style-type: none"> • Launching and Institutionalizing • Funding • Structuring the Organization • Operations • Organizational Planning • Customer Interface Management • Organizational Risk Management • Developing an Acquisition Strategy • Training 	<ul style="list-style-type: none"> • Launching and Institutionalizing • Funding • Structuring the Organization • Operations • Organizational Planning • Customer Interface Management • Organizational Risk Management • Developing an Acquisition Strategy • Training 	<ul style="list-style-type: none"> • Data Collection, Metrics and Tracking • Technical Risk Management • Organizational Risk Management • Customer Interface Management • Organizational Planning

© 2006 Carnegie Mellon University



THE ENTIRE PICTURE



© 2006 Carnegie Mellon University



IN A NUTSHELL

- Software product lines epitomize the concept of strategic, planned reuse.
- The product line concept is about more than a new technology. It is a new way of doing one's software business.
- There are essential product line activities and practices areas as well as product line patterns to make the move to product lines more manageable.



WHAT'S DIFFERENT ABOUT REUSE WITH SOFTWARE PRODUCT LINES?

- Business dimension
- Iteration
- Architecture focus
- Preplanning
- Process **and** product connection

© 2006 Carnegie Mellon University



AT THE HEART OF SUCCESSFUL PRODUCT LINES

- A pressing need that addresses the heart of the business
- Long and deep domain experience
- A legacy base from which to build
- Architectural excellence
- Process discipline
- Management commitment
- Loyalty to the product line as a single entity



FINAL WORD

If properly managed, the benefits of a product line approach far exceed the costs.

Strategic software reuse through a well-managed product line approach achieves business goals for:

- efficiency
- time to market
- productivity
- quality
- agility



**SOFTWARE PRODUCT LINES:
REUSE THAT MAKES BUSINESS SENSE.**



QUESTIONS – NOW OR LATER

Linda Northrop

Director, Product Line Systems Program

Telephone: 412-268-7638

Email: lmn@sei.cmu.edu

U.S. Mail:

Software Engineering Institute

Carnegie Mellon University

4500 Fifth Avenue

Pittsburgh, PA 15213-3890

World Wide Web:

http://www.sei.cmu.edu/productlines/plp_init.html

SEI Fax: 412-268-5758

© 2006 Carnegie Mellon University



Software Engineering Institute

Carnegie Mellon

Software Product Lines: Reuse That Makes Business Sense

Page 74